UNITED STATES DISTRICT COURT
EASTERN DISTRICT OF TEXAS
TYLER DIVISION

UNILOC USA, INC. and UNILOC LUXEMBOURG S.A., Plaintiff,	)	
v.	)	
ALTAIR ENGINEERING, INC. *et al.*, Defendants.	)	Case No. 6:12-cv-806-LED (Consolidated Lead Case)

DEFENDANT'S SUPPLEMENTAL CLAIM CONSTRUCTION BRIEF REGARDING "PROCESS ID"

Pursuant to the Court's bench order during the parties' February 13, 2014 *Markman* hearing, defendant Altair Engineering, Inc. ("Defendant")[1] submits this supplemental brief in support of its proposed construction for the term, "[operating] as an independent process." The parties dispute whether the phrase "process ID" should be included in the construction for this term. Defendant contends that it should be included while Plaintiff proposes it should not. The uncontested evidence before the Court – including the only expert declaration, the only treatises and the inherent nature of CPU operations – all indicates that independent processes inherently each have their own process IDs.

I. "INDEPENDENT PROCESS" = "DIFFERENT PROCESS ID"

Claim 18 requires that the policy server means be operating "as an independent process."[2] The parties agree that the policy server means is a daemon and further agree that a daemon is a computer program. *See* P.R. 4-5 at 4. A computer program is a set of instructions, which exists physically as an inert collection of text and numbers (usually called "code" in the art). *See*

[1] Former defendant SlickEdit, Inc. has recently filed a joint stipulation of dismissal.

[2] *See generally* Defs' Br. (Dkt. 123) at 3-4. "The '222 Patent discloses a particular architecture for performing concurrent licensing…. Claim 18…discloses a daemon, itself a piece of software, that sits on each node and operates as a middle man between the various software products on the node, on the one hand, and their respective network license servers, on the other." *See id.*

generally Nettles Decl. (Dkt. 123-14) at ¶11.[3] Due to its inert nature, a computer program does not execute on its own. *Id.* Rather, a computer must include a special kind of software that enables execution of the computer program's instructions. This special software is called an "operating system." *See* Ex. 1 (Silberschatz) at 1 ("An *operating system* is a program that acts as an intermediary between a user of a computer and the computer hardware…to provide an environment in which a user can execute programs . . . .").[4] Common operating systems include Windows, Mac OSX and UNIX.

An operating system governs a computer's central processing unit (CPU). *See generally* Ex. 2 (Tanenbaum) at 27. To run a computer program, the operating system loads the set of instructions into memory, and the CPU then executes these instructions. *Id.* This *act* of execution is referred to in the art as a *process*.

> The most central concept in any operating system is the process: an abstraction of a running program. . . . The key idea here is that *a process is an activity* of some kind. Ex. 2 (Tanenbaum) at 27, 29 (emphasis added).

As Dr. Nettles explained in his Declaration:

> A "process" is created by a computer's operating system so that an instance of a program can be executed. It is an instance of a single program running on a computer. This is what Silberschatz … was referring to when he wrote:
>
> We emphasize that a program by itself is not a process; a program is a *passive* entity, such as the contents of a file stored on disk, whereas a process is an *active* entity. Nettles Decl. at ¶12 *citing* Silberschatz (Dkt. 123-9) at 90 (emphasis in original).

With respect to the '222 Patent, when the daemon is not running, it exists as an inert

[3] The daemon's code sits/resides/exists "separate" from the software product's code. As discussed at the *Markman* hearing, these traits of (1) physical separateness and (2) independent operation are part-in-parcel of the daemon's ability to service multiple software products.

computer program, maintained locally on the given computer. When the daemon is running, it gives rise to a process, operating on the given computer.

An operating system manages the processes that are running on a computer. Every process, for the purpose of accounting, is assigned a number. This number is often referred to as a "process ID". Dr. Nettles explained this concept in his (uncontested) Declaration:

> Virtually every operating system (including the UNIX system in the '222 Patent specification and Windows and other systems on which the defendants' products operate today) keeps track of the different processes the same way: by giving each process a unique identifier, typically an integer. Different processes have different process identification numbers, or process ID's. Nettles Decl. at 4.

The precise architectures of process management systems vary between operating systems, but the "process" model *requires* operating systems to maintain information about each process running on the computer.[5] As such, each process' corresponding process ID is stored in memory for use by the CPU. *See* Ex. 2 at 32. Without a process ID stored in memory, the CPU has no way to keep track of which process is which, or which program is being serviced. *See* Dkt. 123-9 (Silberschatz) at 94 ("In UNIX, each process is identified by its process identifier, which is a unique integer.").

In short: since every process must have a unique process ID, it follows that the daemon, when operating as an independent process, *must have a unique process ID*. Nettles Decl., ¶19 ("a daemon, operating as a background process, always has its own, unique, process ID"). That is

[4] All Exhibits are attached to the Declaration of Marshall G. MacFarlane in Support of Defendant's Supplemental Claim Construction Brief Regarding "Process ID". Defendant notes that Exhibits 1 and 2 attached here to are additional excerpts of Exhibits attached to Dkt. 123.

[5] Nomenclature varies, but this structure is usually called the Process Control Block (PCB) or process table. *See* Dkt. 123-9 (Silberschatz) at 91-92 ("Each process is represented in the operating system by its own process control block (PCB[) …]A PCB is a data block or record containing many pieces of the information associated with a specific process, including . . . Accounting information. This information includes… job or process numbers").

the very definition of independence. Moreover, because the daemon has a *unique* process ID, it must be one that is different than the software product's process ID.

Uniloc concedes that the policy server daemon is a computer program and operates as an independent process, yet appears to argue that a daemon can have no process ID.[6] Such an argument is nonsensical. Without a process ID, the operating system would have no distinguishable entry for the daemon in memory, could not change the daemon's state (running, terminating), and could not handle inter-process communication between the daemon and the software product ('222 patent, at 5:63).

II. DEFENDANTS' PROPOSED CONSTRUCTION

Defendant proposed a construction in its P.R. 4-2 disclosure: "locally executing instructions with a process ID that is different from the software product's process ID." Based on evidence in the record that is unrefuted and the inherent meaning of the term "independent process" to one of ordinary skill in the art, the Court should adopt this construction.

The phrase "as an independent process" describes *how* the daemon operates. The clause makes clear that, when operating, the daemon must do so on its own, using a distinct process. " *See* Dkt. 123-9 (Silberschatz) at 95 ("A process is independent if it cannot affect or be affected by the other processes executing in the system."). It must operate/run/execute independently. Although process IDs are not explicitly described in the '222 Patent[7], the feature of process IDs

[6] Uniloc has also previously suggested (without support) that a linked library file that has licensing code can qualify as a daemon. Uniloc appears to contend that a library file (which receives calls from a software product and which is also part of the software product) satisfies the "independent process" requirement of Claim 18 when that code begins executing, regardless of whether it has a process ID . This, however, is refuted by the only evidence in the record. As soon as the code in the library file is executed, a process is created and a process ID is assigned. *See* Ex. 2 at ("Processes are named by their PIDs [process IDs]").

[7] It would be unreasonable to expect all characteristics inherent to a feature to be included in a patent specification. For example, one would not always expect to see discussion of "electronic

are inherent to processes. The evidence submitted on the record enforces that one of ordinary skill in the art understands and appreciates process IDs as an inherent feature.

As discussed above, processes are always assigned a process ID. If a daemon is operating as a process, it follows that the daemon must have a process ID. *See* Nettles Decl. at ¶19 ("ever process has … a unique process ID." *citing* Stevens at 72.) And if that daemon is running independently, it also follows that the daemon must have a unique process ID (*i.e.,* one that is different than the software product's process ID.) [8]

III. CONCLUSION

Defendant respectfully requests that the Court adopt its proposed construction to include the inherent feature of "process ID" in its construction of "[operating] as an independent process," which is consistent with the uncontested evidence on record and the understanding of one skilled in the art.

Respectfully submitted,

ALTAIR ENGINEERING, INC.

/s/ Marshall MacFarlane

Marshall MacFarlane
YOUNG BASILE HANLON & MACFARLANE, P.C.
3001 W. Big Beaver Rd., Ste. 624
Troy, MI 48084 / (248) 649-3333
macfarlane@youngbasile.com

Deron R. Dacus
Texas Bar No. 00790553
THE DACUS FIRM, P.C.
821 ESE Loop 323, Suite 430
Tyler, Texas 75701 / 903-705-1117 E-mail:
ddacus@dacusfirm.com

chips" every time a patent described a computer. As known to one skilled in the art, electronic chips are a feature that is simply inherent to computers.

[8] If – despite the un-contradicted evidence in the record – the Court accepts Uniloc's new assertions regarding "processes" that do not have process IDs, the Court can refer the question to the jury for resolution. We respectfully submit, however, that the Court should in no way pronounce that a process can exist without a process ID.

CERTIFICATE OF SERVICE

I, Marshall G. MacFarlane, hereby certify that on February 20, 2014 I caused a true and accurate copy of this document to be served on all counsel of record who consent to electronic service via the Court's CM/ECF system pursuant to Local Rule CV-5(a)(3).

/s/ Marshall G. MacFarlane